Test Generation with Dynamic Impact Analysis for C++ Programs

Mihály Biczó, Krisztián Pócza, and Zoltán Porkoláb

During regression testing tests from an existing test suite are run against a modified version of a program in order to assure that the underlying modifications do not cause any side effects that would demolish the integrity and consistency of the system. Although the test set can grow uncomfortably large, we can safely focus on a smaller subset of modification traversing tests, since only these tests might reveal an error. Unfortunately, in many cases the original and modified system generates identical output for a modification traversing test, which means that the test in question is not effective. In this paper we present an efficient C++ specific approach to automatically repair ineffective test cases. Our solution is a two stage process, both stages performing a forward algorithm, which guarantees results can be obtained on-the-fly. First, the exact cause of ineffectiveness and the affected statement is identified; while in the second stage input variables whose values should be altered to repair the test are selected. Applying our algorithm requires precise instrumentation of the source code. Still this method can be successfully applied to large-scale applications, therefore it not only helps system maintainers to automatically manage the test suite, but it also assures that the system would be run against a harsh test set. Under these circumstances we get a very reliable and safe solution, which minimizes the possibility of an unexpected system failure.

References

[1] B. Korel and A. M. Al-Yami. Automated Regression Test Generation, *ISSTA*, 1998: 143-152

[2] B. Korel. Automated Test Data Generation for Programs with Procedures, *ISSTA* 1996: pp. 209–215

[3] W. Wong, J. Horgan, S. London, and H. Agrawal. A study of effective regression testing in practice, In *Proc. of the Eighth Intl. Symp. on Softw. Rel. Engr.*, pp. 230–238, Nov. 1997. 10

[4] D. Kung, J. Gao, P. Hsia, F. Wen, Y. Toyoshima, and C. Chen. Change impact identification in object-oriented software maintenance, In *Proceedings of the IEEE International Conference on Software Maintenance*, pp. 202–211, September 1994.

[5] B. G. Ryder and F. Tip. Change impact analysis for object oriented programs, In *Proceedings of PASTE '01*, pp. 46–53, Snowbird, Utah, USA, June 2001.

[6] G. Rothermel and M.J. Harrold. Selecting tests and identifying test coverage requirements for modified software, *Proceedings of the International Symposium on SoftwareTesting and Analysis*, pp. 169–184, August 1994.